Fujitsu Microelectronics (Shanghai) Co., Ltd.
Application Note

MCU-AN-500067-E-12

F²MC-8FX FAMILY

8-BIT MICROCONTROLLER

MB95330 SERIES

120° Hall Sensor/Sensorless DC Inverter Control F2MC-8L/8FX SOFTUNE C Library

APPLICATION NOTE

FUJITSU

Revision History

Date	Author	Change of Records
2009-11-20	Kevin Wang	V1.0, First draft
2009-11-30	Kevin Wang	V1.1,Modify
2010-01-11	Kevin Wang	V1.2,Modify

This manual contains 20 pages.

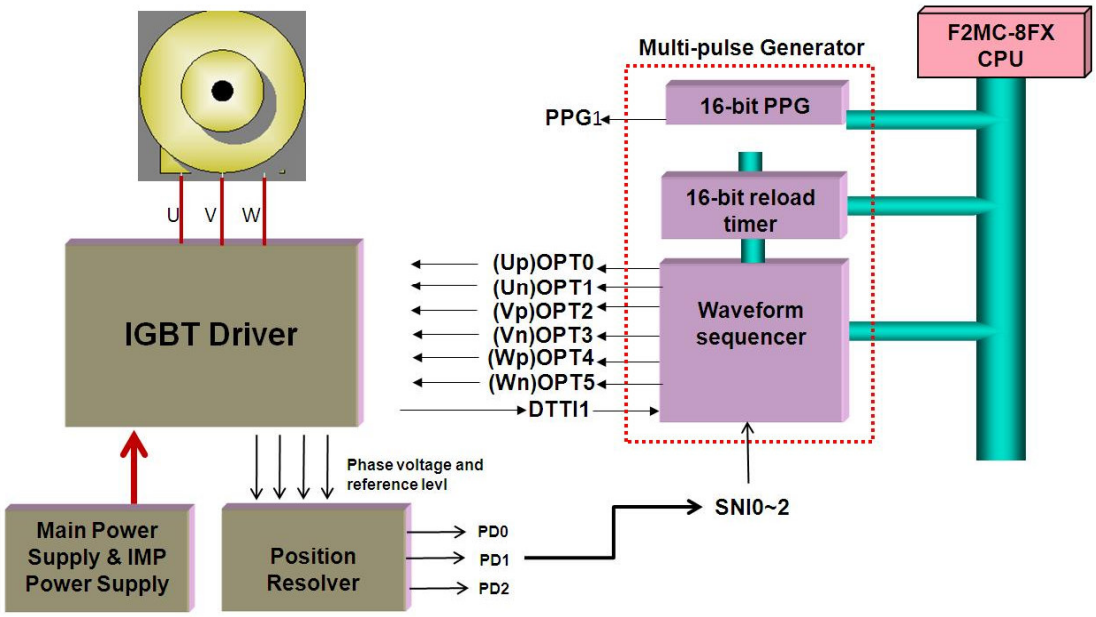
1 Introduction

This document describes the implementation of 120° conduction hall sensor/sensorless brushless DC motor control using the provided F2MC-8L/8FX SOFTUNE C library and the Fujitsu MB95F330 8-bit micro-controller. The operation principles, specification, library installation, library function description and operation of library functions are included. MB95F330 series 8-bit Micro-controller can be used to control the operation of a 3-phase brushless DC motor using the 120° conduction inverter control solution.

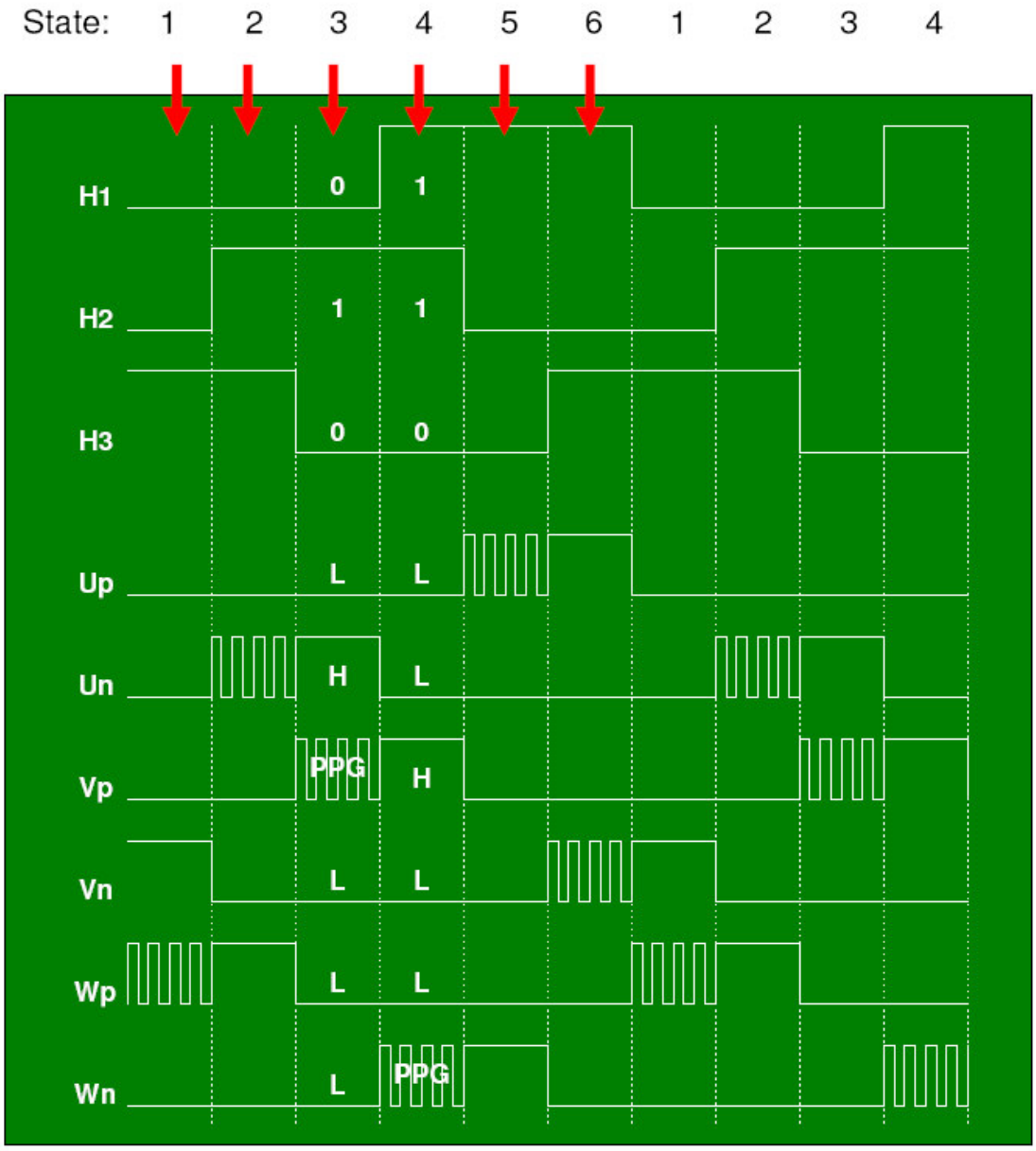
2 Operation Principles and Theory

2.1 Hall Sensor Drive

Below is the brief working principle for MCU to drive motor with hall sensor. A multi-pulse generator outputs six switch signals to drive IGBT inverter. Three channel hall sensor signals are detected by MCU input capture to achieve motor position. One channel over-current signal is output by IGBT inverter to MCU to protect the whole system.



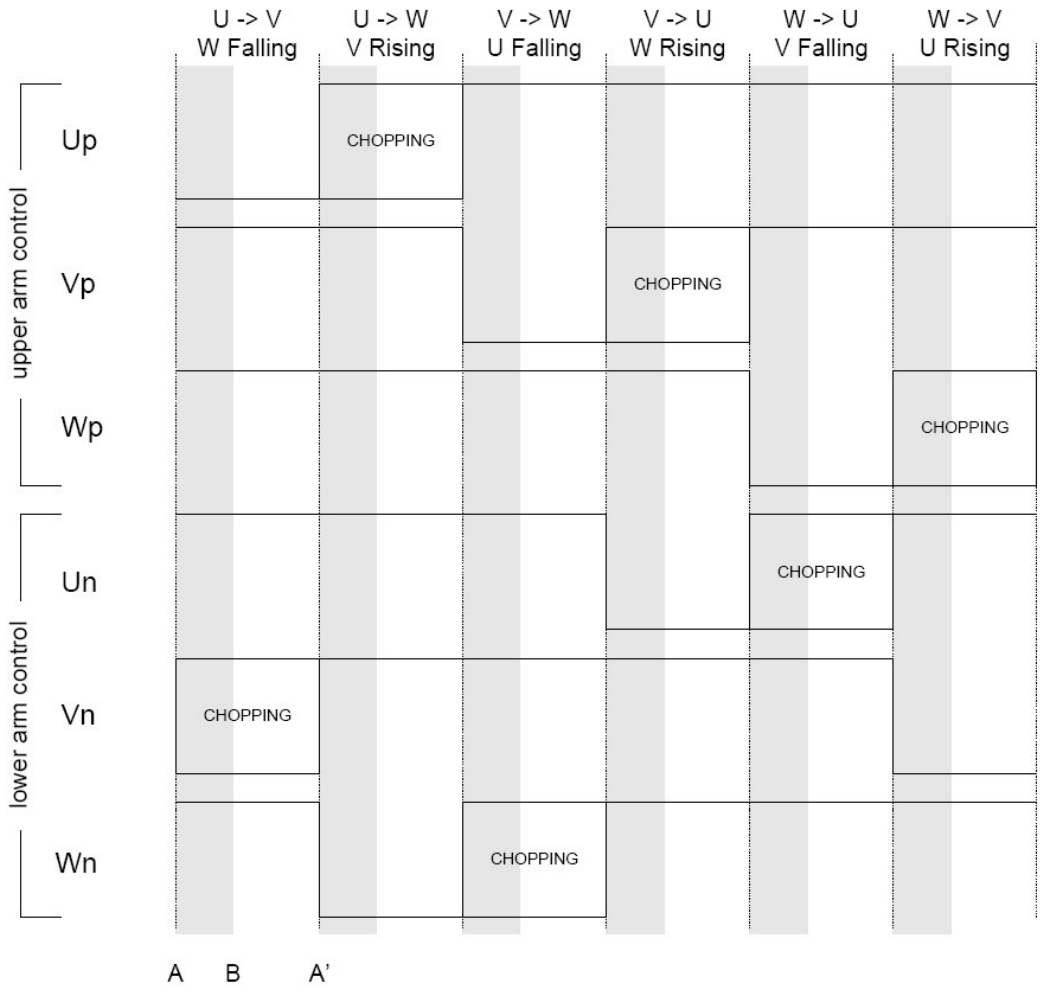
One electrical cycle is divided into 6 states. The relationship between three channel hall sensor signals (H1, H2, H3) and six channel inverter switch signals (Up, Un, Vp, Vn, Wp, Wn) is shown as below:



2.2 Sensorless Drive

2.2.1 Sensorless Startup

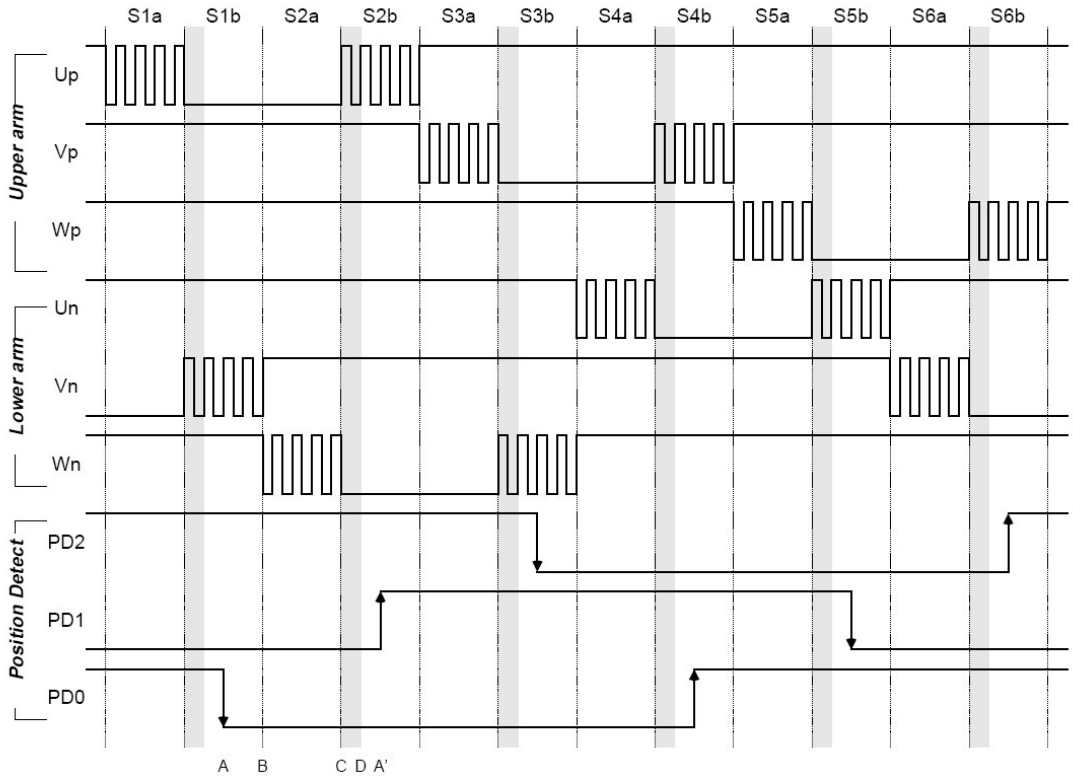
The suggested startup method is forced startup. The following is the driving pattern. The marker A and A' are the state change, while A – B is the position detect mask-off period used to mask off unwanted interrupt when the back EMF is very weak during startup.



2.2.2 Normal Run

The normal run consists of 12 different driving patterns and 6 different states. The following shows the relationship between the driving patterns and the expected interrupts from the position detection circuit.

Marker explanation:
A: position detection interrupt
B: change state
C: change chopping-arm
D: position detection interrupt enable
A': next position detection interrupt
A – B: commutation delay
B –: change arm delay
C – D: change arm mask-off period



3 Library Installation

3.1 Components

The library package contains 3 files:

File name	Usage
motor.lib	Library file, contains all function modules
Motor.h	Header file, contains prototypes of the modules and global variables
myvect.h	Header file, contains the interrupt vector table declaration

3.2 Procedure

There are 3 steps to begin using the Motor.lib C library.

- ✓ In F2MC-8L/8FX SOFTUNE, after creation of a new project, use PROJECT –> ADD MEMBER to add motor.lib as a member.
- ✓ Include Motor.h header file into C main program for external references.
- ✓ Include myvect.h header file into the module which uses directive #pragma to generate the interrupt vector table.

Thus, a project including Lib file is ready for the caller program.

4 Library Functions and External Variables

There are 4 global variables in the library:

- ✓ Rotation_Direction
- ✓ Start_Motor
- ✓ Driver_Mode
- ✓ Motor_State

There are 8 functional modules for library control:

- ✓ Motor_Init,
- ✓ Sensor_Less_Start
- ✓ Motor_Parm
- ✓ Motor_Set_Change_Speed
- ✓ Motor_Stop
- ✓ Sensor_Less_Normal_Work
- ✓ Hall_Sensor_Start
- ✓ Hall_Sensor_Normal_Work

4.1 Function Syntax

Syntax	extern void Motor_Init(void);
Description	Initialize MCU resources to be ready for start and stop commands. • Initialize port configuration. • Initialize multi-function timer resources. • Initialize speed check timer. • Initialize interrupt. • Initialize motor state to MOTOR_READY.
Input parameters	Void
Return	Void

Syntax	extern void Sensor_Less_Start(unsigned short start_duty_on, unsigned short start_period, unsigned short normal_duty_on, unsigned short normal_period);
Description	Start motor from reset with sensorless drive • Start_motor will be MOTOR_READY. • Startup and normal run parameters are initialized.
Input parameters	start_duty_on : startup carrier frequency duty on duration in 125ns unit Start_period : startup carrier period in 125ns period unit Normal_duty_on : carrier duty on duration when startup changes to normal run, in 125ns unit normal_duty : carrier period in normal run mode
Return	Void
Example	Sensor_Less_Start(400, 1600, 200, 800); 60us on time during startup = 400 x 125ns => 60000 5kHz carrier frequency => 1600 x 125ns startup carrier period, 25us on time just after startup = 200 x 125ns => 25000 10kHz carrier frequency => 800 x 125ns normal run carrier period

Syntax	extern void Motor_Parm(unsigned long speed_con, unsigned short csd, unsigned short cad,unsigned short camaskt, unsigned short stmaskt);
Description	Define runtime parameters with sensorless drive. • Define speed constant for speed checking • Define commutation delay duration • Define the duration between change-state and change-arm • Define the mask-off period just after change-arm • Define the mask-off period during startup
Input parameters	speed_con= 60 / (2us x number of pole pair) csd, in x100 electric angle cad, in x100 electric angle camaskt, in x100 electric angle stmaskt, in 1us unit
Return	Void
Example	Motor_Parm(15000000, 0, 200,200, 2000); 2 pole pair => 60 / (2us x 2) = 15000000 0 change state delay after back EMF zero crossing => 0 2 change-arm delay after back EMF zero crossing => 200 After change arm, mask time =>200 During startup, 2ms = 2000 x 1us => 2000

Syntax	extern void Motor_Set_Change_Speed(unsigned short speed);
Description	Set or change target rotational speed in RPM whenever sensorless drive or hall sensor drive is used.
Input parameters	speed in RPM
Return	Void
Example	Motor_Set_Change_Speed(6000); Set target speed to 6000rpm.

Syntax	extern void Motor_Stop(void);
Description	Stop motor without brake. • All driving outputs are inactivated. • Speed checking timer is stopped. • Multi-function timer is reset. • Input capture edge detection are disabled.
Input parameters	Void
Return	Void

Syntax	extern void Sensor_Less_Normal_Work(void);
Description	Control motor running normally with sensorless drive. • Count change arm time.
Input parameters	Void
Return	Void

Syntax	extern void Hall_Sensor_Start(unsigned short duty_on, unsigned short period);
Description	Start motor from reset with hall sensor drive. • Start_motor will be MOTOR_READY. • Parameters are initialized
Input parameters	duty_on :Carrier frequency duty on duration in 125ns unit period : Carrier period in 125ns period unit
Return	Void
Example	Hall_Sensor_Start (150, 800); 18.75us on time during startup = 150 x 125ns => 150 10kHz carrier frequency => 800 x 125ns startup carrier period,

Syntax	extern void Hall_Sensor_Normal_Work(void);
Description	Control motor running normally with hall sensor drive. • Count motor speed. • Control motor speed. • Check hall sensor signal and change arm.
Input parameters	Void
Return	Void

4.2 External Variables

Variable	extern unsigned char Motor_State
Description	Motor operation mode
Value	MOTOR_READY, 1 : motor ready for accepting start command MOTOR_START, 2 : motor in startup stage MOTOR_NORMAL, 3 : motor in normal run stage MOTOR_FAILURE, 4 : motor which cannot run

Variable	extern unsigned char Rotation_Direction
Description	Motor running direction
Value	ANTICLOCKWISE, 0: motor anticlockwise running CLOCKWISE, 1: motor clockwise running.

Variable	extern unsigned char Driver_Mode
Description	Motor drive method
Value	HALL_SENSOR, 0: hall sensor drive SENSOR_LESS, 1: sensorless drive.

Variable	extern unsigned char Start_Motor
Description	Start motor signal
Value	FALSE, 0: the motor cannot be started. TRUE, 1: the motor can be started.

5 Usage of Library Functions

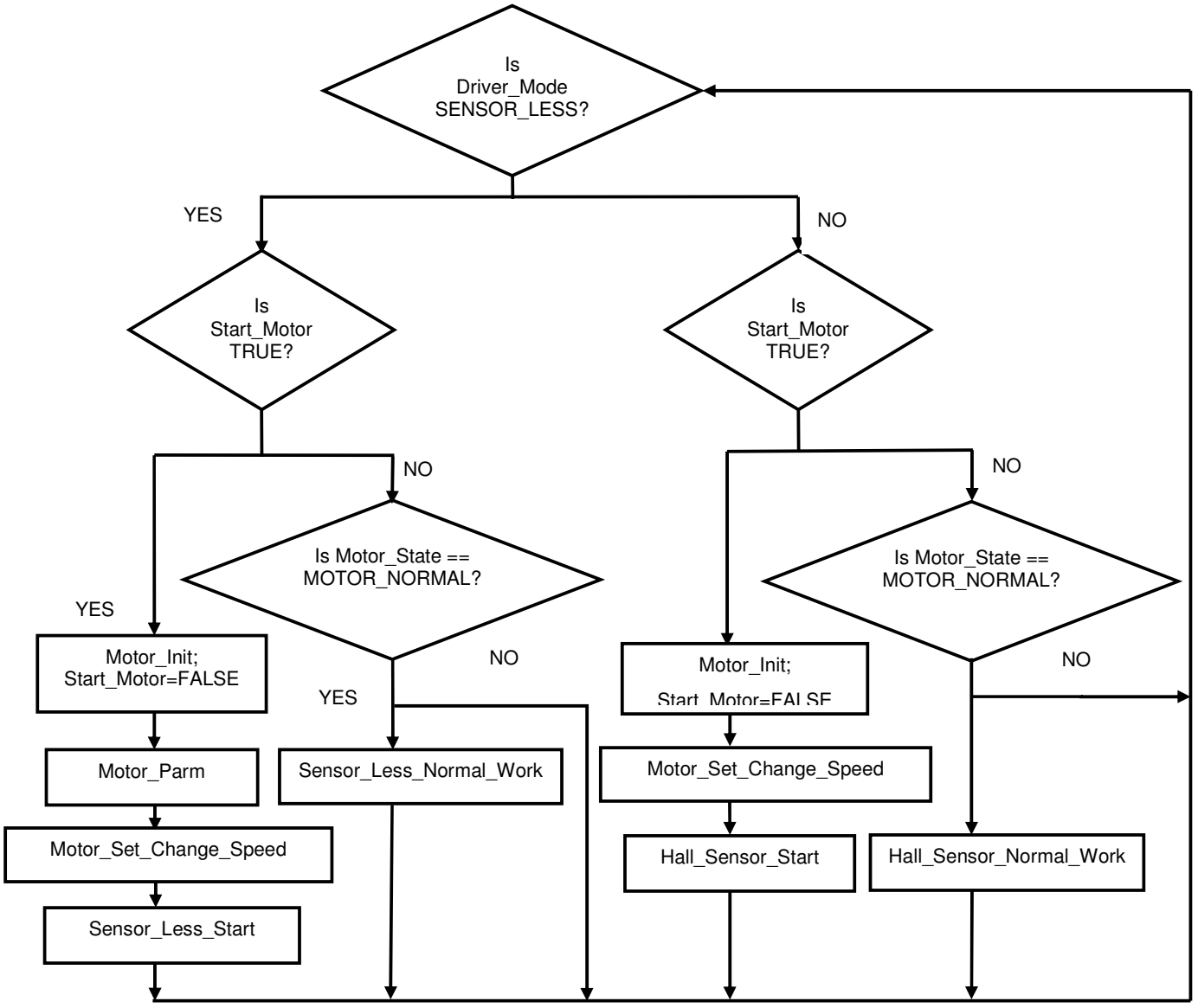
In general，user should follow the following steps to control the motor:

- ✓ Set global variables with suitable values.
- ✓ Initialize the MCU resource.
- ✓ Start the motor with suitable startup speed.
- ✓ Modify motor synchronous speed, accelerating speed and decelerating speed by changing values of the global variables.
- ✓ Stop the motor.

5.1 Operation Flow

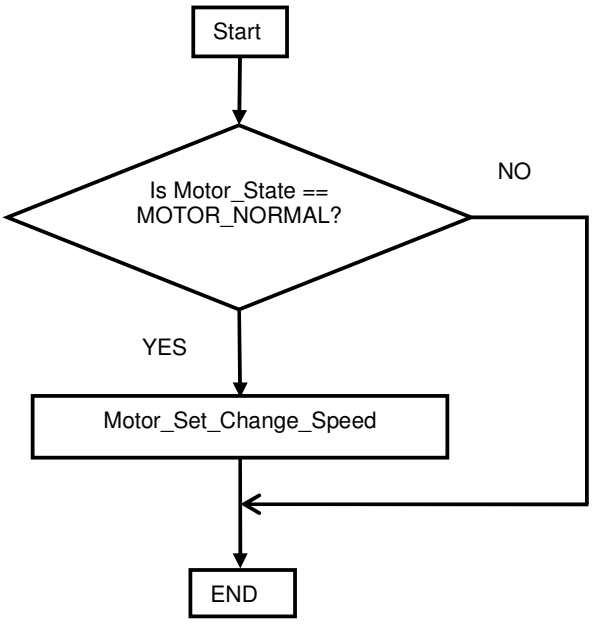
5.1.1 Start Motor

This can be done by calling the following successively using appropriate parameters.



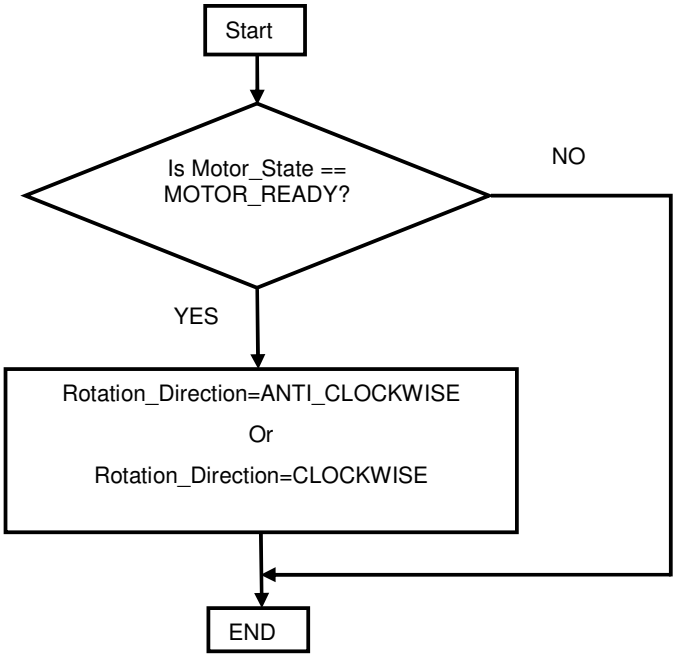
5.1.2 Change Motor Speed

To change motor speed, please ensure that the motor is running under normal status. The following flow chart shows how to change the motor speed:



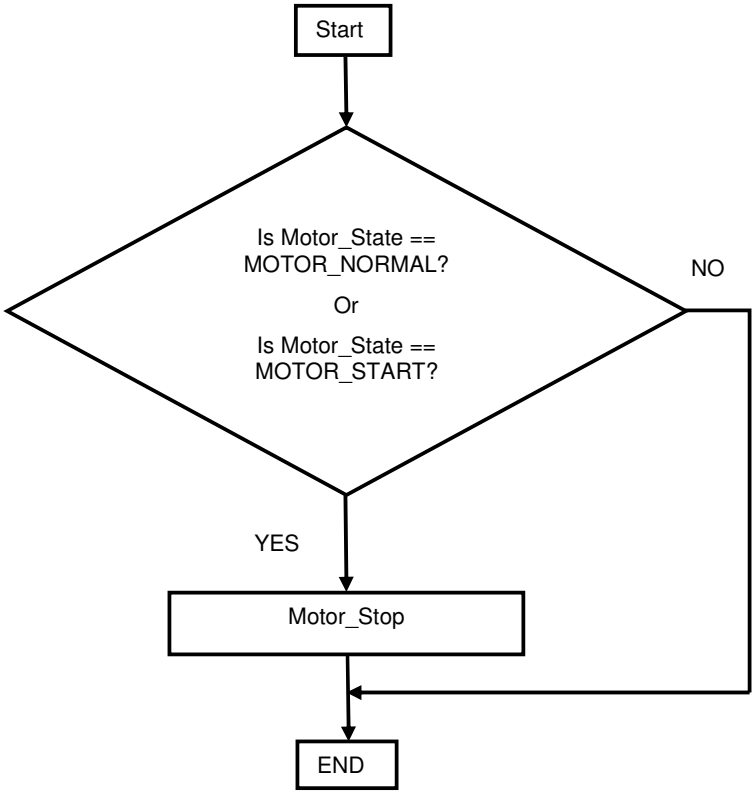
5.1.3 Set Motor Rotation Direction

To set motor rotation direction, please ensure that the motor is under ready status. The following flow chart shows how to set the motor rotation direction.



5.1.4 Stop Motor

To stop a motor, please ensure that the motor is under normal or startup status. The following flow chart shows how to stop the motor.



6 Sample Program

Motor.zip is a sample project containing source code which can drive a sensorless brushless or hall sensor DC motor with motor EV Board (PN: MB2146-440-E V1.2). Please refer to Motor EV Board MB2146-440-E HW User Manual.
Tested configuration:
DC motor: Fulling FL28BL26-15V-8006AF
Number of phases: 3
Number of poles: 4
Supply voltage: 15VDC
Minimum tested speed: 1000rpm
Maximum tested speed: 7000rpm
MCU work load: 8%~30% (Motor speed from 1000 rpm to 7000 rpm with sensorless drive); 2%~10% (Motor speed from 1000 rpm to 7000 rpm with hall sensor drive);

7 Additional Information

For more information on how to use MB9595330 EV Board, BGM adaptor and SOFTUNE, please refer to Motor EV Board MB2146-440-E HW User Manual or visit

Website: http://www.fujitsu.com/cn/fmc/services/mcu/


FUJITSU